The Shoppers Super Store

Software Development Project

**Requirement Specifications**

# Overview

This document outlines the requirements to build a software application in Java 1.8 to simulate some of the fundamental shopping experiences encounter when you go to your favorite multi-purpose super store.

# Goal

The goal of this exercise is to give students the opportunity to learn the use of various Java sdk 1.8 classes using real world experience they are already familiar with.

# Software Development Lifecycle

This project will use the Agile lifestyle development approach to build the Shoppers Super Store application. We will work in one-week sprints, which means each developer will be assigned a task(s) to complete in one week.

We will have scrum (stand up) meets at the beginning of every time we meet in Zoom. During stand up each developer will report the follow:

* What did you accomplish since our last scrum?
* What are you currently working on today?
* Do you have anything blocking you from moving forward? (Blockers)

**Scrum Team**

A scrum team will be made up of a small group of developers working towards a common goal. The members of the team are:

* Scrum Master – Oversees each Scrum (standup) and Guide the Agile Process
* Product Owner – Reviews and Approves all deliverables
* Developers – Software Development Engineers

# Project Software Structure

The Shoppers Super Store (S3) project software components will be stored in standard file hierarchy structure:

C:\usi-git\SoftwareEngineeringTraining\projects\src\main\java\com\usi\shopperssuperstore\

Packages

Each department in the S3 Store will represent a package. Here are some examples

* bakery
* customer service
* produce

This is where your java files will exist.

# Design and Requirements

Together as a development team we will define the requirements for the store and each department. It will be the developer’s responsibility to create the necessary classes to fulfill the requirements of each department they are assigned based upon their collective shopping experience.

**Customer Service Dept**

* ***Super Store Class* –** is responsible for store maintenance. This is the class that is the application. From this class all other departments will be managed.
  + Is each department ready for business Hint: Y/N?
  + There must be away to close a department.
  + There must be away to close the store.
* ***Greeting*** ***Class*** – Must keep track of Customer name and Membership Card. If the customer has a membership card on file (in a text file), then they must get the membership discount.
  + Needs to be put into a txt file in the resource package.
  + Needs to keep track of Name, Phone Number, Email address, ~~Mailing Address~~?, DOB, etc.
  + Need Customer Class
  + Need Membership Card Class – Gold, Silver and Bronze
* ***Shopping Cart (****Buggy****) Class*** – the shopping cart must provide a way to load products into the cart and keep a running total on it’s Super Store Shopping Cart LED.
  + - Shopping simulation
* ***Checkout Class*** – the checkout class must tally the items in the Shopping Cart and tender the transaction. It must be able to receive cash, Visa, Master, and EBT. When tendering cash, this method must return correct change. Please apply taxes where appropriate.
  + Define a way to apply taxes according to state.
  + Apply taxes based upon product type, i.e… food vs Not.
  + Do we ask if they want their receipt printed or emailed? Or we make that part of the membership information? Should there be an override option?
  + “has-a” Receipt object defined as a data member.
* ***Receipt*** ***Class*** – The customer can choose to get a printed receipt or have a copy emailed to them.
  + There is currently a ***StorePrinter*** class (which works in all 50 states), which can be used by the Receipt class.
  + Must be able to receive an array of products that on in the ‘basket’.
* Reports
  + Should we do reports for Departments, Purchases for the day, Profit and Loss, Sales by Departments.

**Departments**

* ***Record Loading*** – Each department subclass is responsible for loading and maintaining a list of records according to their unique truck loads.
* ***Product Loading*** – Each department subclass is responsible for building an HashMap of products using the previously loaded records.
* ***Allocating Isles*** – Each department subclass will be responsible for obtaining and maintaining a set of Isles from the Super Store. It must also provide the ability to provide a list of products, currently on their department isles.
  + **Isle class exist**
* ***Load Shelves*** – Each Department subclass will be responsible for loading the shelves of each Isle with products in their department.
* **Shelve Class** -- exist

**Product Classes**

* ***Product Content Label*** -- Each product subclass must have a method called to String() which will print a neatly formatted vertical list of all of the pertinent information a customer would want to know about a product. This method will be invoke whenever you reference the instants name by the Java compiler.
* ***Product Price*** – there must be a method getPrice() that returns the price of each product.